Computer Vision Lab 2: Camera Calibration

Robin Khatri, M1 MLDM

March 2019

Contents

1 Introduction

We know that the 3D world co-ordinate of a point is related to the image co-ordinate of the point on image plane in 2D as follows:

$$U = MP \Rightarrow U = K_s K_f \pi_0 P' \tag{1}$$

Where P' is 3D point in camera coordinates and P is the world point. P' is produced from point P by multiplying it with a matrix of extrinsic parameters consisting of eucledean transformations in 3D space. Here U is the scaled image point of point P (world co-ordinate). Both U and P are presented as homogeneous co-ordinates in equation 1. M is a 3 by 4 matrix and it contains extrinsic and intrinsic parameters. Intrinsic parameters - focal length f of camera, resolution, skew factor, distance of points O_x and O_y (The correspondence from camera coordinates to image coordinates) known as image center, distortion etc. - are related to camera's characteristics. Extrinsic parameters are related to camera's position and pose (where the camera is and how it is oriented). In three dimensional Computer Vision, it is important to have knowledge about intrinsic parameters as it helps in generating 3D co-ordinates of image points and it also helps in removing lens distortion and in increasing accuracy.

The process of recovering intrinsic (and/or extrinsic) parameters from images is called camera calibration.

During the practical, we learnt about camera calibration using Camera calibration Toolbox from Caltech developed in MATLAB (Bouguet [2008]). We went through two procedures - Calibrating a single camera and calibrating a stereo system - with the help of two examples provided at `http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html`.

2 Calibrating a Single Camera

Use of chessboards:

As mentioned in previous section, we need some images for camera calibration. We chose these images to be chessboard reflecting different orientations. Chessboards are the most popular input for calibration because their corners are simpler to detect and also relatively, the corners are less invariant to distortions.

From `http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html`, we obtained images (total 20) which were included in the first example. Following set of pictures were obtained:

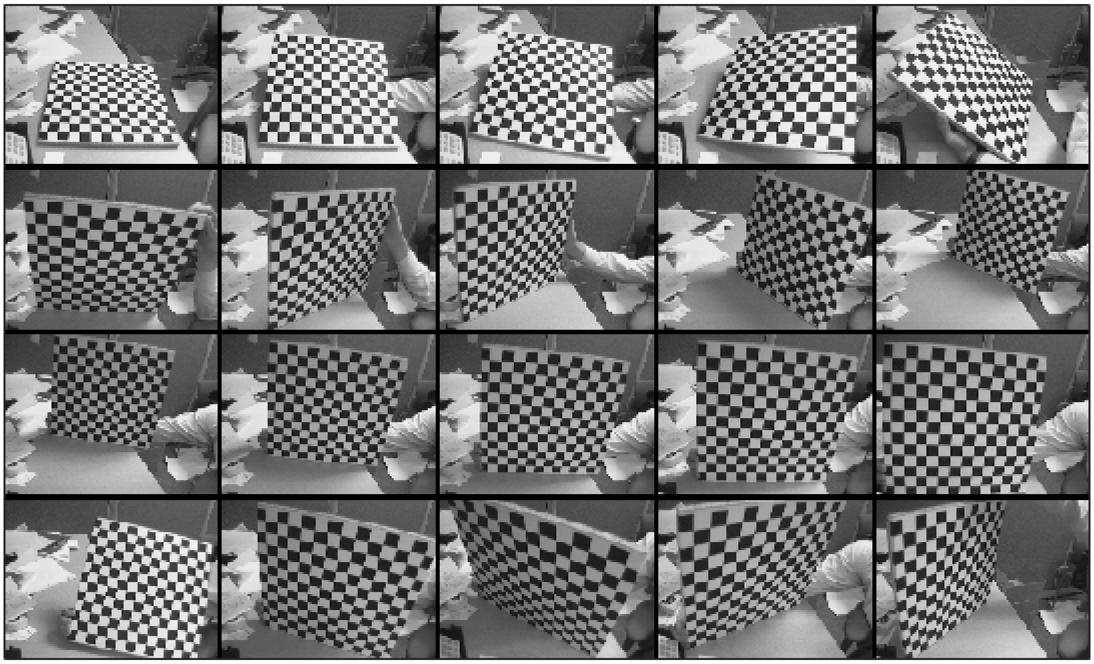

Figure 1: Chessboard Images

Images shown in figure 1 are taken from the a single camera and that is the camera we wish to calibrate.

2.1 Corner Extraction

To extract corners from our chessboard images, we selected 4 extreme corner points from every image. This process was manual. We used a window size of 1111 pixels to find corners. The idea is very simple here. When we select four extreme corner points and also provide values of d_x and d_y (which is 30 mm by default in the toolbox and can be changed depending on the image), the toolbox can easily find all the squares between these 4 extreme corner points.

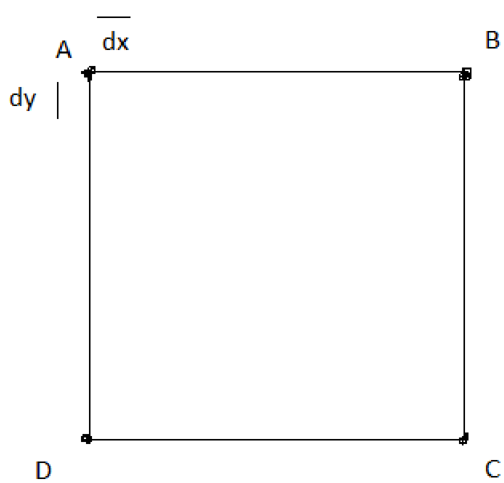

Figure 2: Extreme Corners (A,B,C,D) and supplied d_x and d_y

It is easy to see that after supplying four extreme corner points, the camera knows the co-ordinates of these points. When we also input values of d_x and d_y, computation of number of squares between A, B, C and D is easy. Further, possible corner points between black and white squares of our chessboard images can be calculated in a simple manner.

Note:

1. As noted, d_x and d_y are important values to input as they measure the number of squares in between extreme points and thus they approximate finding corners. Because of this, sometimes due to lens distortion the prediction of corners may not be accurate.

2. The first extreme corner that is selected is considered a reference point for image frame. On multiple images, we need to be careful to select the same grid corner as reference as the image plane co-ordinates will be dependent to that.

3. As clearly seen, the extreme corners are highly important in initializing procedure of camera calibration so they need to be selected carefully and a large error (in terms of pixels) should not be made.

4. During single camera calibration in our lab session, we did not consider skew and assumed the angle between x and y axes on image plane to be 90 degrees.

On this basis, we selected the extreme corner points in all 20 images. For Image2.tif, the extreme corners selected are presented in figure 3. Figure 4 shows the identification of corners from the toolbox using the technique as defined above.

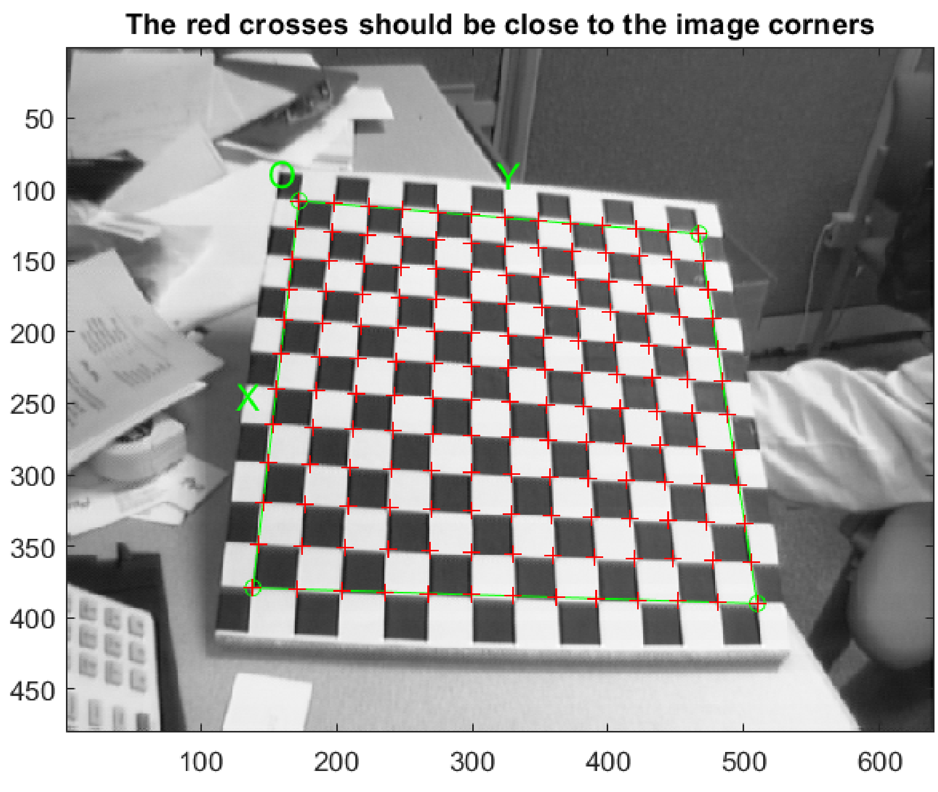

Figure 3: Selected Extreme corners - Image2.tif

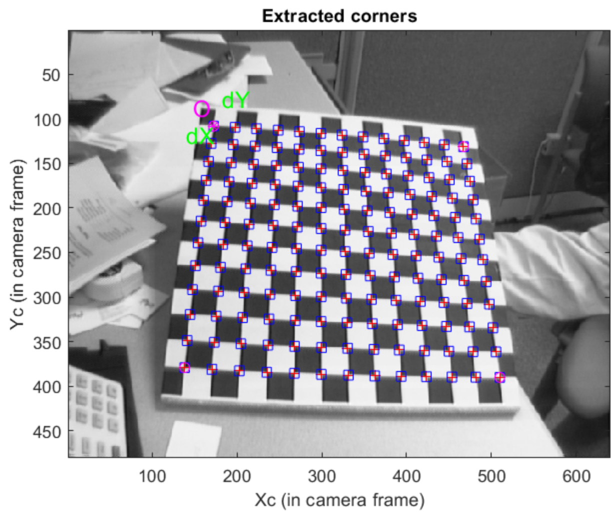

Figure 4: Extracted corners - Image2.tif

As mentioned in note 1 above, we may need to account for lens distortion to accurately identify corners.

Recall: If (X_c, Y_c, Z_c) are the 3D co-ordinates in camera co-ordinates then the corresponding pinhole projection can be given by:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \end{bmatrix} \tag{2}$$

Now, define $r = \sqrt{(x^2 + y^2)}$,

Adding lens distortion, the transformed point would be,

$$\begin{bmatrix} x_d(1) \\ x_d(2) \end{bmatrix} = (1 + k_c(1)r^2 + k_c(2)r^4 + k_c(5)r^6) \begin{bmatrix} x \\ y \end{bmatrix} + D(x) \tag{3}$$

Where D(x) is tangential distortion vector. In equation 3, k_c is a vector containing tangential and radial distortions. Its first coefficient can be used to identify corners correctly to account for lens distortion. Its value ranges between -1 and 1. So, to deal with distortions, in the Calibration toolbox, we can select a value between -1 and 1 to account for distortions while extracting corners. On images 15, 16 and 18, this was used. we set first coefficient stored in k_c to be -0.3 for image 15, -0.1 for images 16, 18, 19 and 20 and -0.2 for image 17.

2.2 Calibrating after extracting corners

We know that to recover parameters, we need to solve the following optimization problem:

$$\begin{aligned} A\mu &= 0 \\ argmin_{\mu} ||A\mu|| &= 0 \\ s.t. ||\mu|| &= 1 \end{aligned}$$

Where μ is the 12 dimensional vector consisting of 12 coefficients of 3 by 3 matrix M given in equation 1. A is the matrix consisting of linear combinations of points in image plane and in camera co-ordinates. For every point selected we get 2 equations. So, our first step was to solve this optimization problem. Doing so, we obtained following results:

```
Calibration parameters after initialization:

Focal Length:          fc = [ 667.17198   667.17198 ]
Principal point:       cc = [ 319.50000   239.50000 ]
Skew:             alpha_c = [ 0.00000 ]   => angle of pixel = 90.00000 degrees
Distortion:            kc = [ 0.00000   0.00000   0.00000   0.00000   0.00000 ]
```

After doing so, we need to do non-linear optimization for optimized values of instrinsic parameters. From equation 2, we saw that $(x_d(1), x_d(2))$ were the image coordinates after taking into account distortion. If we denote pixel coordinates by (x_p, y_p), the relationship between these can be given by following transformation of their homogeneous co-ordinates.,

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = K \begin{bmatrix} x_d(1) \\ x_d(2) \\ 1 \end{bmatrix} \tag{4}$$

Where K is the camera matrix given by,

$$K = \begin{bmatrix} f(1) & \alpha f(1) & O_x \\ 0 & f(2) & O_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Where f is the focal length of camera; f(1) is the focal length expressed in units of horizontal pixels and f(2) is the focal length expressed in units of vertical pixels.

This matrix is used in re-projecting and finding estimated values. Then we can easily calculate error at every point i (u_i, u_j) and then taking their distribution into account, we can arrive at their optimized values. The amount of time it takes to arrive at the non-linear optimization solution is dependent upon our initial guess. The closer our initial guess is to the original values, the faster the convergence will be. The output of this non-linear optimization procedure is given below:

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 658.02421   659.34344 ] +/- [ 0.90889   0.97395 ]
Principal point:       cc = [ 300.98601   243.27040 ] +/- [ 1.85213   1.68771 ]
Skew:             alpha_c = [ 0.00000 ] +/- [ 0.00000  ]   => angle of pixel axes
                              = 90.00000 +/- 0.00000 degrees
Distortion:            kc = [ -0.26426   0.18793   -0.00000   -0.00030  0.00000 ]
                            +/- [ 0.00710   0.02834   0.00039   0.00039  0.00000 ]
Pixel error:          err = [ 0.42048   0.20399 ]
```

As noted in Note (4) in section 2.1, skew distortion coefficients were not calibrated and so we have them as default at 90 degrees.

2.3 Re-projection through optimized parameters

After our parameters have been optimized, we can see their correctness visually by re-projecting corners in our chessboard images with values defined by our optimized parameters from section 2.2.

In figure 5, the detected corners represented (from section 2.1) and re-projected corners can be seen for Image18.tif.

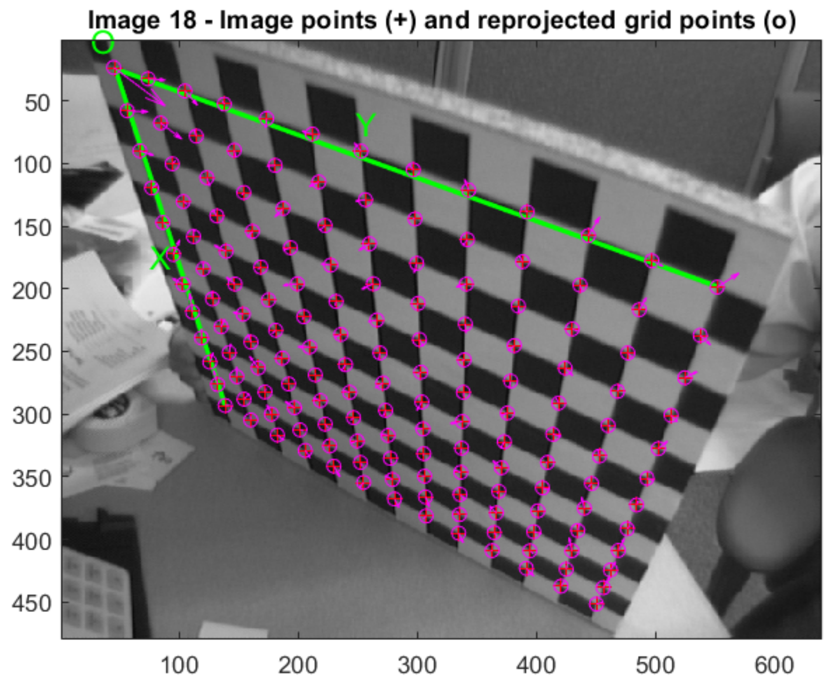

Figure 5: Re-projection of grid corners

The error between the original and projected space is given below. Figure 6 represents the re-projection errors in x-y pixel plane. Blue represents the pixel co-ordinates of predicted corners while red points are extracted corners.

```
Number(s) of image(s) to show ([] = all images) =
Pixel error:      err = [0.42048   0.30399] (all active images)
```

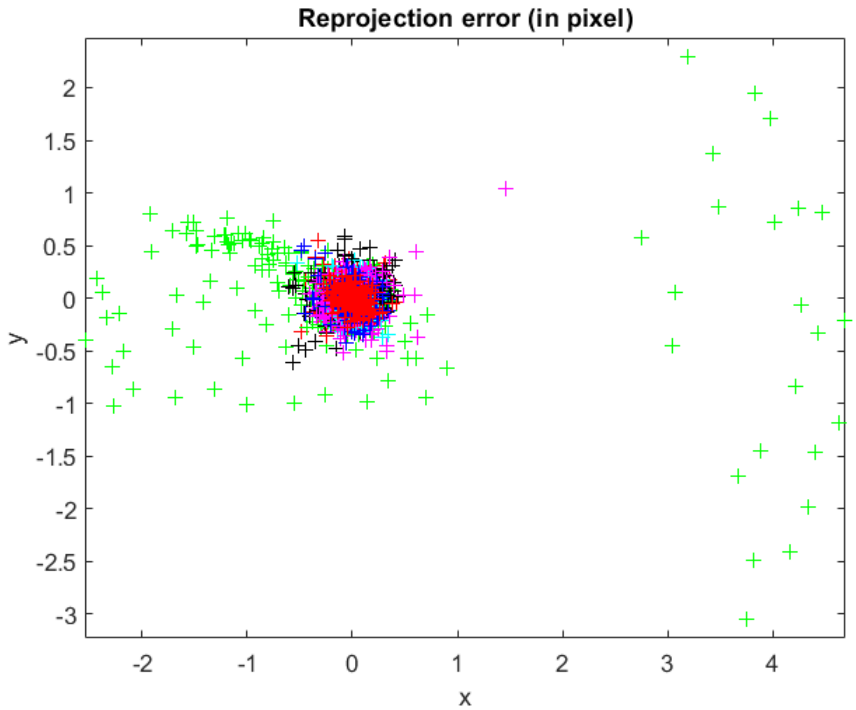

Figure 6: Error - Re-projected corners(section 2.2) - Blue vs. detected corners (section 2.1) - Red

Recovered extrinsic and instrinsic parameters can also be represented in 3D as shown in figures 7 and 8.

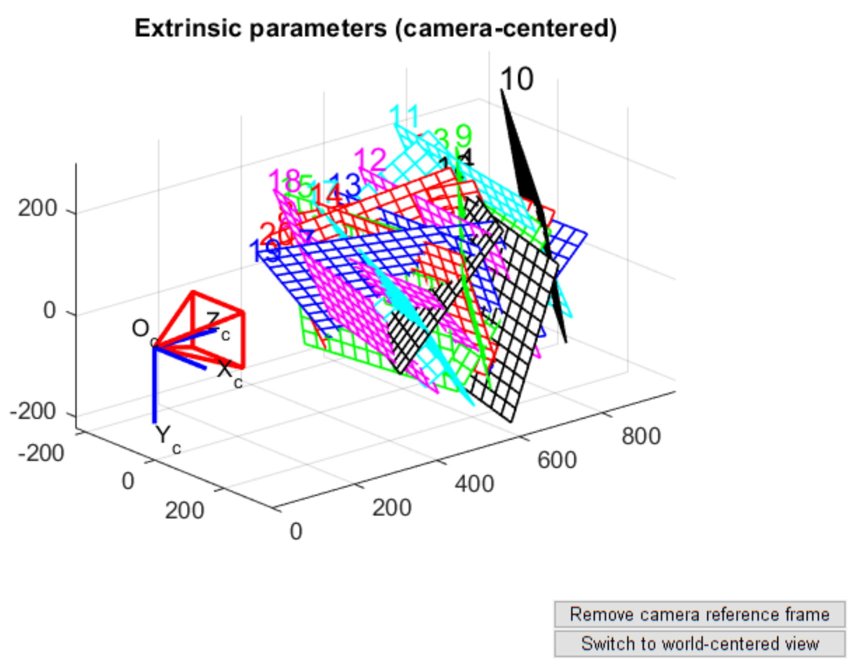

Figure 7: 3D view of extrinsic parameters (Camera-centered)

"Camera-centered" simply means seeing the object (Chessboard) (and associated intrinsic parameters) like camera sees i.e. camera co-ordinates are fixed (relatively speaking). "World-centered" means that we could see different orientations of camera.

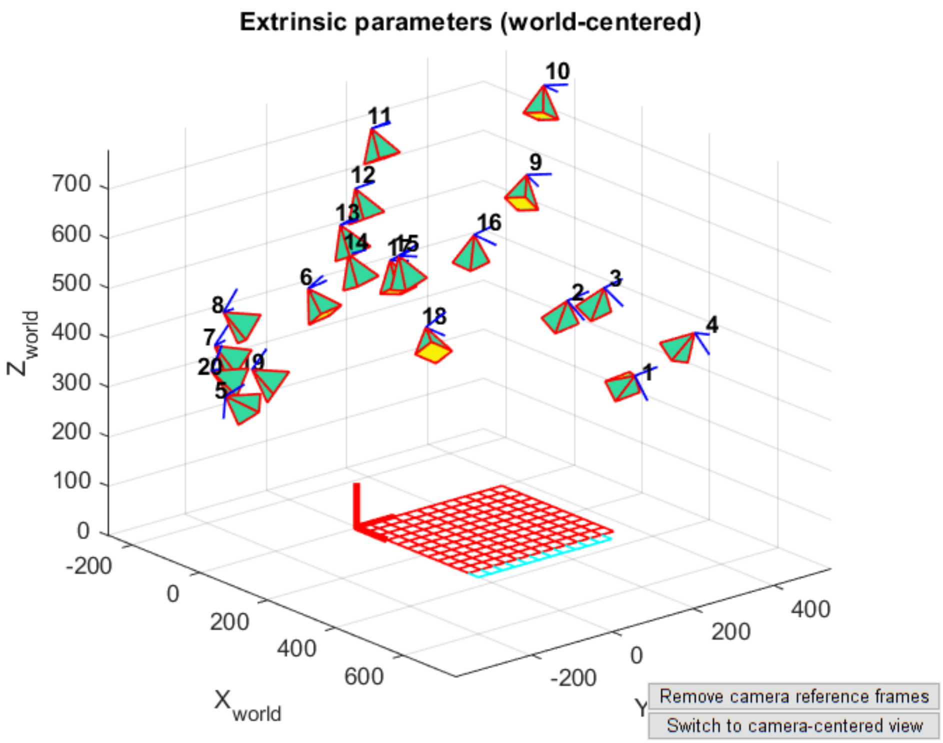

Figure 8: 3D view of extrinsic parameters (World-centered)

2.4 Dealing with errors

As stated in note 4 of section 2.1, we did not take into account skew and distortions. We did input the first coefficient for distortion vector k_c as mentioned in section 2.2. but we did not do it on all images and perhaps not with a lot of accuracy, that is the reason why in figure 6, we can see a disparity between re-projected and detected corners. This can be mitigated by using the Re-computation feature of Calibration toolbox. What that does is - it adds predicted distortions to remap the points and thus approximates parameters to a great accuracy. After doing this, we can calibrate as before. This results in following output and error for parameters:

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 657.39215   657.76063 ] +/- [ 0.34680   0.37099 ]
Principal point:       cc = [ 302.97777   242.61890 ] +/- [ 0.70524   0.64532 ]
Skew:             alpha_c = [ 0.00000 ] +/- [ 0.00000  ]
                            => angle of pixel axes = 90.00000 +/- 0.00000 degrees
Distortion:            kc = [ -0.25584   0.12758   -0.00021   0.00003  0.00000 ]
                                +/- [ 0.00271   0.01075   0.00015   0.00014  0.00000 ]
Pixel error:          err = [ 0.12664   0.12601 ]
```

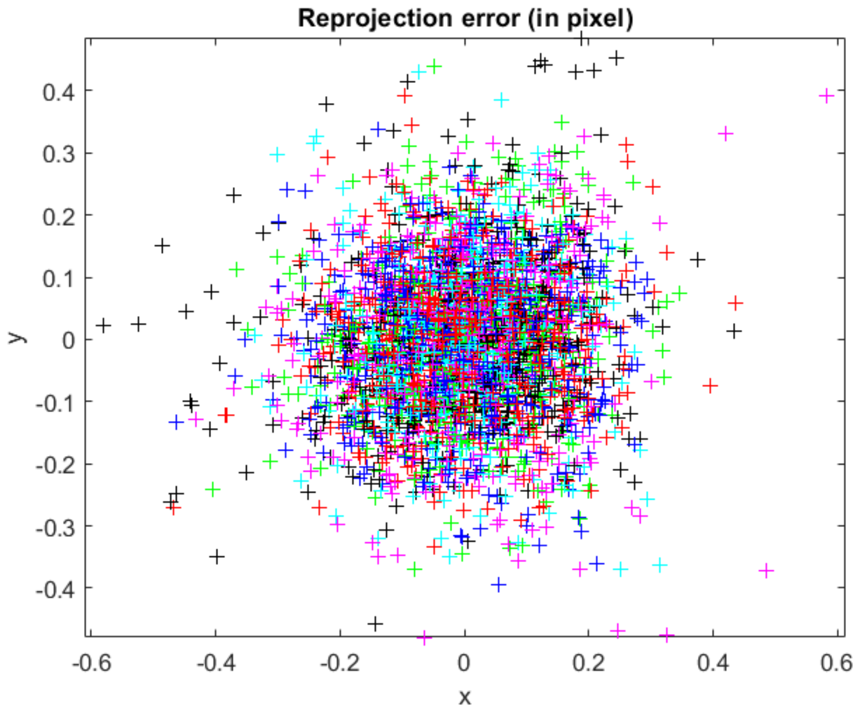

Figure 9: Reprojection error

2.4.1 Note on Selecting distortion models

As detailed in previous section, recomputation (recomp.) of the calibration tool looks at the distortion model to accurately minimize error between reprojected corners and detected corners. The calibration tool provides a way to visualize distortion models. However, the computed distortion vector k_c (defined in section 2.2) should take into account the aspect ratio $\frac{f(1)}{f(2}$ (see section 2.2). However when we get our distortion model from the visualizations provided by the calibration toolbox, it does not take into account the aspect ratio (which is basically the ratio of focal length expressed in horizontal and vertical pixel units). Therefore such a method to approximate parameters is not highly accurate.

2.5 Extrinsic Parameters

Every image taken corresponds to a different camera pose, and the information about this camera orientation can be recovered for a new image given we have already computed intrinsic parameters.

We used an external image for which the extrinsic parameters were computed, the procedure is based on equations defined in section 2.1 with some arithmetic manipulation. The result is presented in figure below.

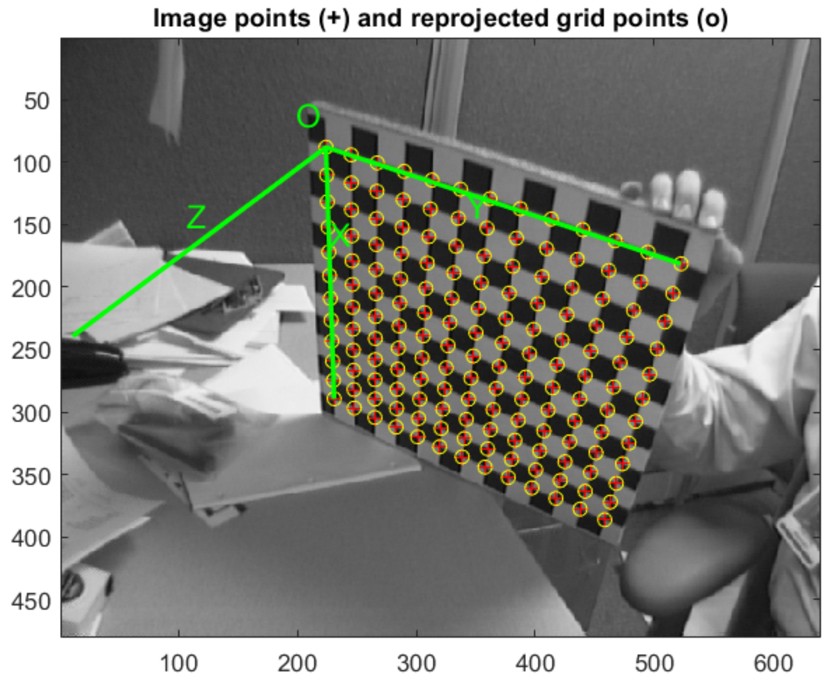

Figure 10: Caption

2.6 Undistorting images

Now since we know the intrinsic parameters associated with our camera, it is easy to see that we can utilize these parameters in getting undistorted versions of images. For Image19, both initial and undistorted versions are shown below:

3 Stereo Calibration

In calibrating stereo system, our purpose is to calibrate two cameras together and recovering extrinsic parameters, and the correspondence points. This procedure also helps in image rectification. Same scene is viewed from two different points, we can call them left and right. In figure 9, we see how this setup works:

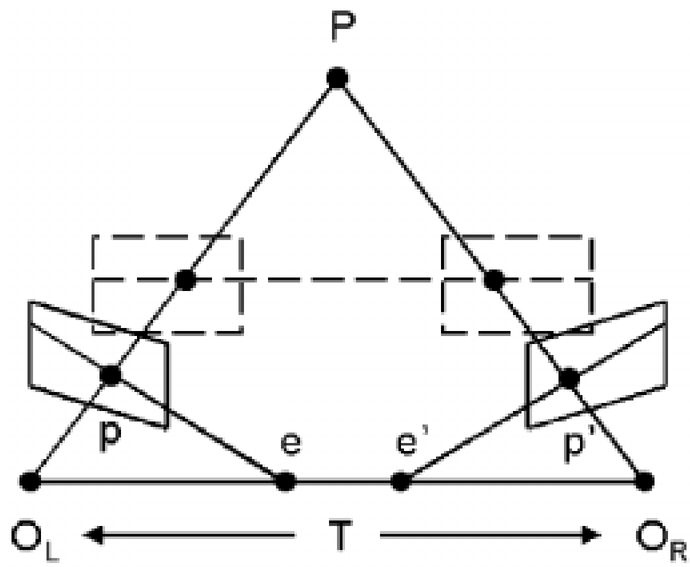

Figure 11: Stereo Setup

Here O_L and O_R are left and right positions of camera respectively. Same object point P (in 3D) is viewed and two images are captured from two different views. Points p and p' in image plane are the points of correspondence. The line connecting O_L and O_R is referred to as baseline (Vector T in image), and the points where base line intersects with image plane are called "epipoles". If we draw a line connecting point p with epipole e then we get epipolar lines. These epipolar lines are crucial in "matching" process i.e. correspondence of same points in two images from two points of view. Our goal is to align these epipolar lines and turn them into so-called scan lines. Then we shall be restricted to do only a 1 D search to find corresponding point of p in the other image because the point corresponding to p will lie on line Ep where E is essential matrix.
We worked with 28 images in this lab session (14 for left view and 14 for right view). These images come from example 5 of Caltech Calibration toolbox `http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example5.html`. An example pair of images is presented in figure 11.

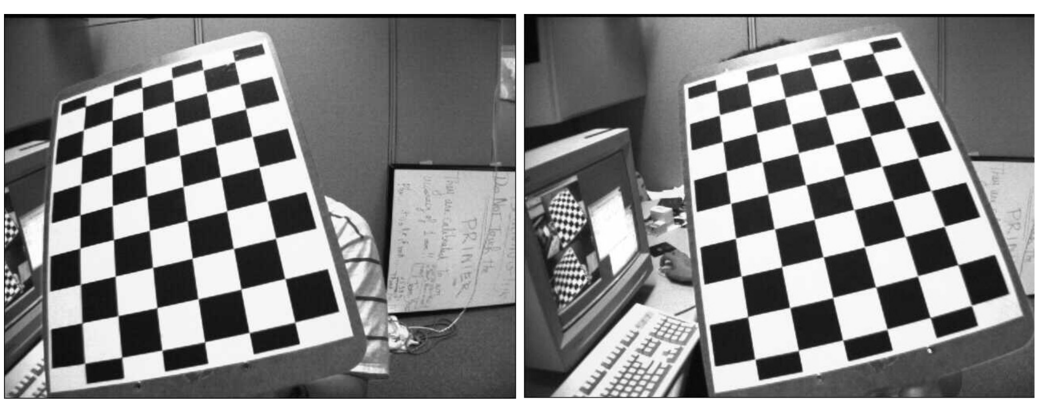

Figure 12: Left and right views of Image5.jpg

We have seen in section 2 how to calibrate a single camera. Following the same procedure, we recovered extrinsic and intrinsic parameters for both left and right set of images.

Extrinsic parameters of right view can be calculated with respect to left camera with center at O_L. Recall that we can relate them by using transformation as represented by figure 12.

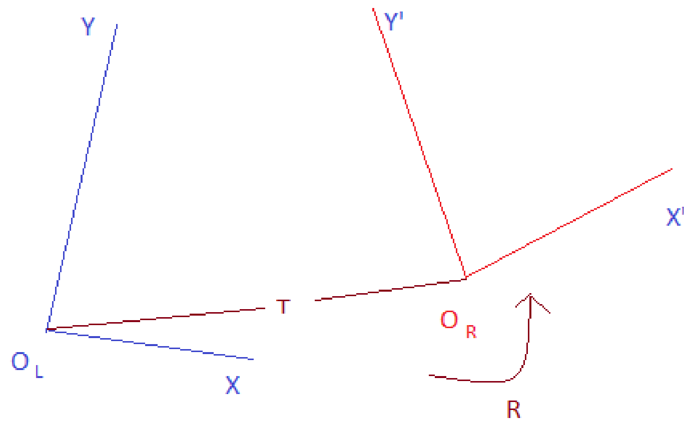

Figure 13: Transformation from O_L to O_R

The equation can be given by,

$$O_R = RO_L + T \tag{6}$$

If we were to transform one point in right frame system to left frame system, we need to consider the skew-symmetric matrix associated with the cross-product of rotation and translation matrix, represented by $t_{[x]}$.

The formula to convert rotation vector to this formulation is also referred to as rodrigues formula (Dai [2015]).

Now suppose the co-ordinates of points p and $p^{'}$ are X_L and X_R. These two points are image of same 3D point P. In calib toolbox, these points are represented by, $X_R = RX_L + T$, same as equation 6.

Because there is always some noise or disparity between two views. Different pair of points may result in slightly different values for rotation and translation. In stereo calibration, goal is to estimate intrinsic and extrinsic parameters from two views where both views can be related (as per equation 6 and through correspondence) with each other and with the 3D world-point while minimizing the re-projection errors for both camera views. This is opposed to single camera calibration, where to estimate parameters we minimized the error of extracted and re-projected corners.
Result of this process is are optimized both extrinsic parameters (R and T) and also previously single-camera-calibrated intrinsic parameters.

The result of calibration for stereo calibration are given below:

```
Intrinsic parameters of left camera:

Focal Length:          fc_left = [ 533.52331   533.52700 ] +- [ 0.83147   0.84055 ]
Principal point:       cc_left = [ 341.60377   235.19287 ] +- [ 1.23937   1.20470 ]
Skew:             alpha_c_left = [ 0.00000 ] +- [ 0.00000  ]
                                 => angle of pixel axes = 90.00000  +- 0.00000 degrees
Distortion:            kc_left = [ -0.28838   0.09714   0.00109   -0.00030  0.00000 ]
                                 +- [ 0.00621   0.02155   0.00028   0.00034  0.00000 ]


Intrinsic parameters of right camera:
```

```
Focal Length:          fc_right = [ 536.81376   536.47649 ] +- [ 0.87631   0.86541 ]
Principal point:       cc_right = [ 326.28655   250.10121 ] +- [ 1.31444   1.16609 ]
Skew:             alpha_c_right = [ 0.00000 ] +- [ 0.00000  ]
                                => angle of pixel axes = 90.00000 +- 0.00000 degrees
Distortion:            kc_right = [ -0.28943   0.10690   -0.00059   0.00014  0.00000 ]
                                +- [ 0.00486   0.00883   0.00022   0.00055  0.00000 ]


Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ 0.00669   0.00452  -0.00350 ]
+- [ 0.00270   0.00308  0.00029 ]
Translation vector:           T = [ -99.80198   1.12443  0.05041 ]
+- [ 0.14200   0.11352  0.49773]
```

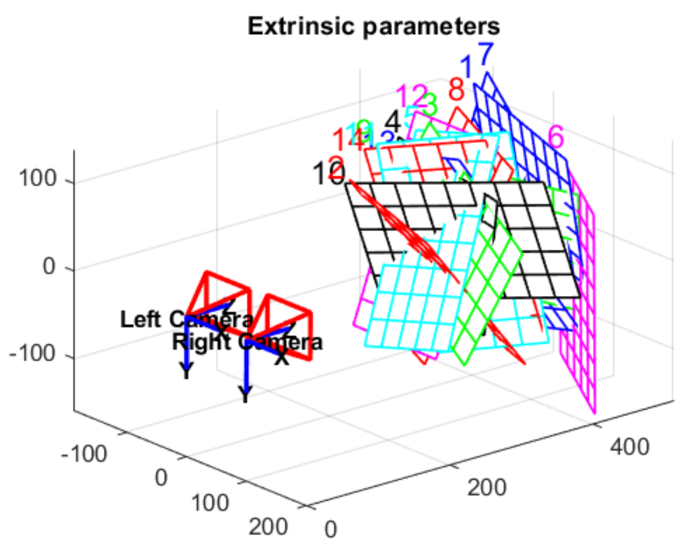

Figure 14: Extrinsic Parameters - Stereo Calibration

All the parameters are already defined, except om. In Caltech calibration set-up, om is a rotation vector and as discussed above it is related to Rotation matrix R through skew-symmetric matrix $t_{[x]}$ (See equation 6).

4 Image Rectification

Simply put, it is the process of searching for correspondence in 2D to 1D through the application of epipolar geometry. We have already seen definitions of epipolar lines and their relation in finding scan lines (for search) in section 3. It is essentially a geometry transformation which transform a camera configuration with non-paraller epipolar lines into canonical one, i.e. colinear epipolar lines.. The process is similar to the algorithm given in 2000 by Fusiello et al. [2000]. Since our cameras are calibrated, we use the obtained parameters for image rectification as opposed to estimating fundamental matrix. Basic procedure can be defined as follows:

1. Camera Calibration to get Intrinsic/Extrinsic Parameters through camera calibration (sections 2 and 3)
2. We compute a rotation matrix R_r (r for rectified) with a purpose to rotate our left camera with center O_L to make epipole e go to infinity - thus making epipolar lines horizontal.
3. Repeat step 2 for right camera.
4. In the transformed stereo set-up, we can calculate location of point (for every point p in left image)
5. Repeat step 4 for right camera.
6. Convert point co-ordinates from camera co-ordinate to pixel image-plane co-ordinates
7. For each pixel in the rectified image, we can find corresponding pixel in the original image (for both left and right). This step is similar to step 4 and 5 but we are going in opposite direction here. This ensures that there are no holes in the process, and helps in minimizing error and maximizing information.

The procedure was conducted automatically in MATLAB through the Calib toolbox. The results are shown below for image 14.

Figure 15: Rectified image14 (right and left views)

5 A short note on stereo triangulation

From points p and p' in figure 10, recovering the 3D co-ordinates of world point P is referred to as stereo triangulation. Using the left and right co-ordinates of points (i.e. co-ordinates of correspondence points), we estimated the location of 3D points. We generated 3D location of the point P with respect to left and right frames of reference. Since these left and right co-ordinates are related by equation 6, we also looked at how close we can get to the left 3D co-ordinate by recomputing it from the right 3D co-ordinate, and they matched well. This is an assurance regarding the accuracy as well as correct recovery of the 3D world co-ordinates.

References

Jean-Yves Bouguet. Camera calibration toolbox for matlab (2008). *URL http://www. vision. caltech. edu/bouguetj/calib_doc*, 1080, 2008.

Jian S Dai. Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections. *Mechanism and Machine Theory*, 92:144–152, 2015.

Andrea Fusiello, Emanuele Trucco, and Alessandro Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.